Was That Even English? Language Modeling for QA

Dan Salo

Duke University

June 22nd, 2017

Outline

Preliminaries

Language Model Types

- N-Grams
- (Word2Vec)
- Recurrent Neural Network
- Convolutional Neural Network

State-of-the-Art

Go Humans Go

Aoccdrnig to a rseearch sduty at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be in the rghit pdae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.

Figure: Humans 1, Computers 0

Motivation

Figure: With what probability is this document written in correct English?

- **Contextual Synonyms**
  $p(\textit{high}$ crowd tonight$) <$
  $p(\textit{large}$ crowd tonight$)$
- **Contextual Spelling**
  $p(\textit{principal}$ scorer$) <$
  $p(\textit{principle}$ scorer$)$
- **Grammar Checking**
  $p($He *play* well$) <$
  $p($He *plays* well$)$

Bag of Words

Word	Count
the	10
ball	7
said	5
court	4
minutes	2
...	...
minuets	1
minuts	1
Curry	1
Game (5)	1

Figure: Frequency of words in document

Advantages:

- Subject Matter Information (ball, court)
- Dictionary Methods (minuts)

Limitations:

- No Contextual Information!

Chain Rule

Sentence probability = joint probability of word sequence:

$$S = (w_1, w_2, w_3, \ldots, w_n) \Rightarrow p(S) = p(w_1, w_2, w_3, \ldots, w_n)$$

Joint probability = conditional $\times$ marginal probability:

$$\begin{aligned} p(w_1, w_2, w_3, \ldots, w_n) &= p(w_1) \times p(w_2|w_1) \times p(w_3|w_1, w_2) \ldots \\ &\quad \times p(w_n|w_1, w_2, \ldots w_{n-1}) \\ &= \prod_i p(w_i|w_1, w_2, \ldots, w_{i-1}) \end{aligned}$$

Approximations to true sentence probability:

True	Single, Bag of Words	Double, Markov	Three
$p(S)$	$p(w_i)$	$p(w_i\|w_{i-1})$	$p(w_i\|w_{i-2}, w_{i-1})$

Context Example

As the player shot the __A__ with two __B__ to go in the game, the crowd __C__ in amazement.

A. ball, gala, basketball (Synonyms)

B. minutes, minuets, minutea (Spelling)

C. stares, started, stared (Grammar)

Probabilities:

A. $p(\text{ball}|\text{As, the, player, } \mathbf{shot}\text{, the})$

B. $p(\text{minutes}|\text{As, the, player, } \mathbf{shot}\text{, the, ball, with two})$

C. $p(\text{stared}|\text{As, the, player, } \mathbf{shot}\text{, the, ball, with two, minutes, to go, in, the, game, ”,” ,the, crowd})$

N-Grams:
Language Modeling with Discrete Phrases

N-Grams

Idea: word probability based on phrase frequency in large corpus:

$$p(w_i|w_{i-1}) = \frac{p(w_i, w_{i-1})}{p(w_i)} \approx \frac{\text{count}(w_{i-1}, w_i)}{\text{count}(w_{i-1})}$$

Unigram:	p(ball)	count("ball")
Bigram:	p(ball | the)	$\frac{\text{count("the ball")}}{\text{count("the")}}$
Trigram:	p(ball | shot, the)	$\frac{\text{count("shot the ball")}}{\text{count("shot the")}}$
4-gram:	p(ball | Curry, shot, the)	$\frac{\text{count("player shot the ball")}}{\text{count("player shot the")}}$
5-gram:	p(ball | the, player, shot, the)	$\frac{\text{count("player shot the ball")}}{\text{count("the player shot the")}}$

Questions:

- What about unseen phrases?
- How do we combine all these probabilities?

Smoothing for Unobserved N-Grams

Problem: $p(\text{stared}|\text{the, crowd}) = 0$ because "the crowd stared" was not found in the training corpus.

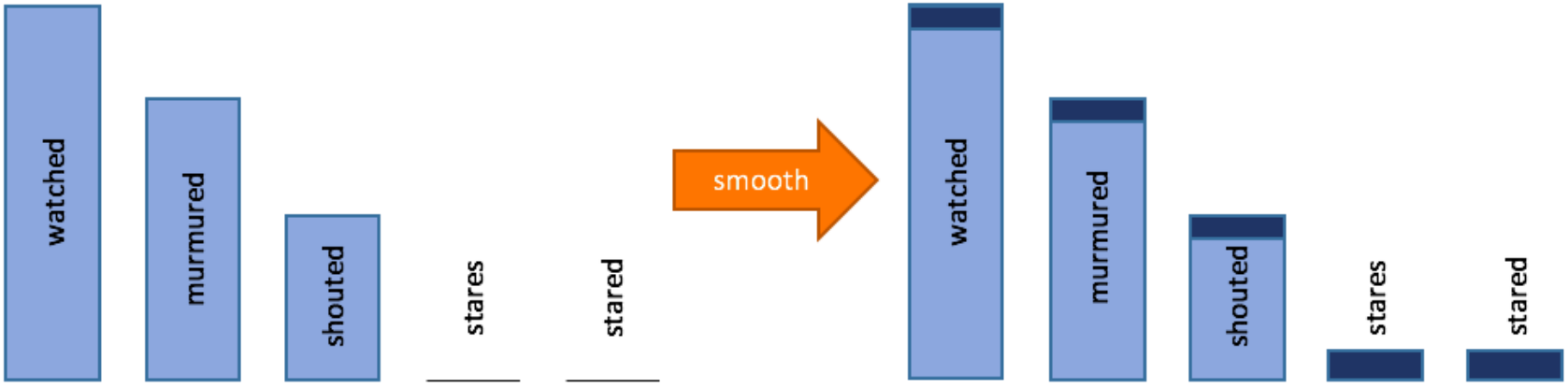

Figure: Steal from the rich and give to the poor.

Idea: Borrow mass from observed n-grams, $p(\text{watched}|\text{the, crowd})$

- Add 1 or a constant to all counts (LaPlace)
- Take mass from n-grams observed 10 times and assign mass to n-grams observed 9 times, etc. (Good-Turing)

Interpolation and Backoff Combine N-Grams

Interpolation: Linear Model of N-grams with learned constants. Works better on smaller datasets.

$$p(\text{stared}|\text{the, crowd}) \approx \lambda_3 \times p(\text{stared}|\text{the, crowd}) + \lambda_2 \times p(\text{stared}|\text{crowd}) + \lambda_1 \times p(\text{stared})$$

Backoff: Use highest N-gram possible. If not above threshold, try a lower order n-gram. Works better on larger datasets.

$$p(\text{stared}|\text{the, crowd}) = \begin{cases} p(\text{stared}|\text{the, crowd}), & \text{if count} > k_{\text{trigram}} \\ p(\text{stared}|\text{crowd}), & \text{if count} > k_{\text{bigram}} \\ p(\text{stared}) & \text{otherwise} \end{cases}$$

Google N-Gram Dataset

4-Gram	Count
serve as the incoming	92
serve as the incubator	99
serve as the independent	794
serve as the index	223
serve as the indication	72
serve as the indicator	120
serve as the indicators	45
serve as the indispensable	111

Figure: Sampling from Google N-Grams

- Released in 2006 as open source
- Compiled from 1T words, 13M unique words
- Comphrensive and still widely used (i.e. LangTool)

N-Grams: Limited Context

As the player shot the ___A___ *with two* ___B___ *to go in the game, the crowd* ___C___ *in amazement.*

A. ball, gala, basketball (Synonyms)

B. minutes, minuets, minutea (Spelling)

C. stares, started, stared (Grammar)

Probabilities (from Google N-Grams):

A. $\frac{p(\text{ball}|\text{shot, the})}{p(\text{gala}|\text{shot, the}}) = 33.7 \Rightarrow$ **ball**

B. $\frac{p(\text{minutes}|\text{ball, with two})}{p(\text{minuets}|\text{ball, with two})} + \frac{p(\text{go}|\text{minutes, to})}{p(\text{go}|\text{minuets, to})} = 30.7 \Rightarrow$ **minutes**

C. $\frac{p(\text{stared}|\text{the, crowd})}{p(\text{stares}|\text{the, crowd})} \approx 1 \Rightarrow$ **???**

Word2Vec:
Continous Word Representations with Semantic Meaning

One-Hot Word Representation

Word	One-Hot
the	$[1000\ldots00]$
ball	$[0100\ldots00]$
said	$[0010\ldots00]$
court	$[0001\ldots00]$
...	...
Curry	$[0000\ldots10]$
minuets	$[0000\ldots01]$

Figure: 'ball' and 'said' are closest in distance but not meaning.

Advantages:

- Easy and Fast

Limitations:

- No Semantic Information!
- Distance carries no meaning.
- Adding new words is difficult.
- Binary, unsmooth loss.

Use Neural Networks to Learn a Continuous Representation!

Neural Networks: What?

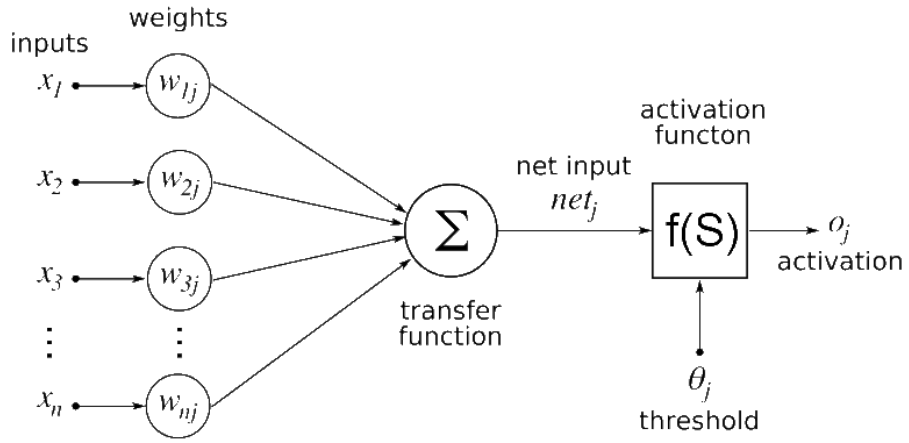

Figure: One Layer of a Neural Network

Steps:

- Multiply inputs and weights $(x_j * w_{nj})$ and add results.
- Apply non-linearity (ReLU: if $< 0, = 0$).

Training:

- Compute loss at network top
- "Backpropagate" to network bottom and update weights.

Neural Networks: Why?

- Biologically-Inspired
- Matrix Multiplication or Dimensionality Reduction
- Data-Driven Features (Not Handcrafted)
- Now? More data. More computational power.
- Universal Function Approximator a.k.a. the Black Box:

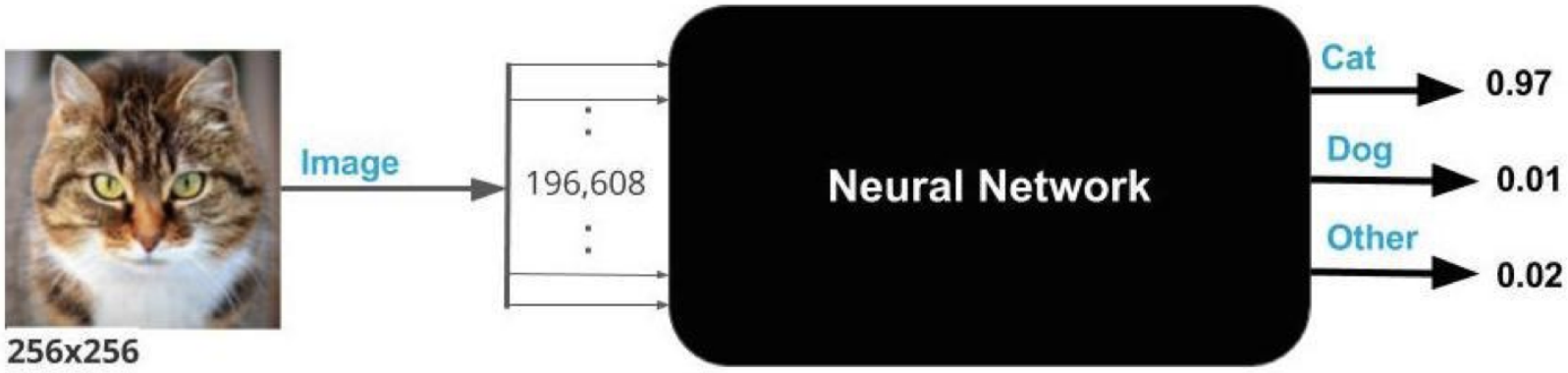

Figure: But Can It Be Trusted?

Word2Vec Model

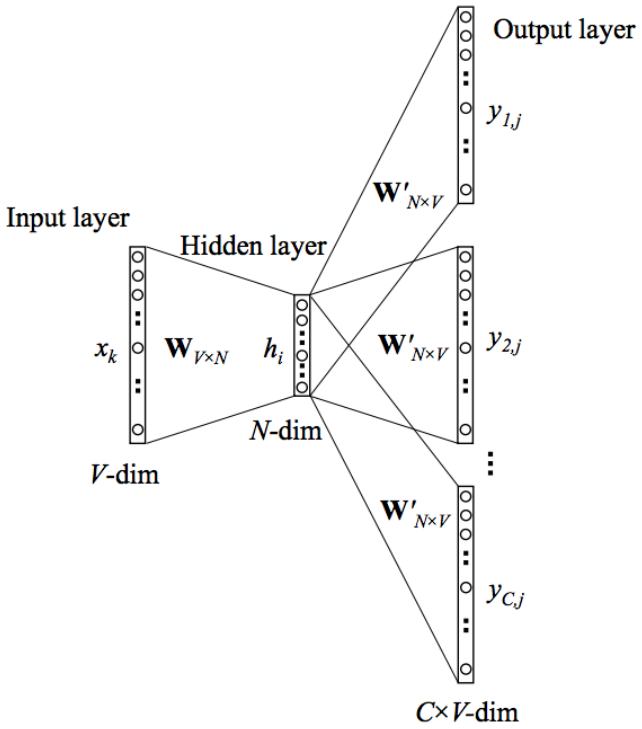

Figure: Word2Vec Model: Neural Network with one Hidden Layer

Advantages:

- Continuous word representation
- Compound words become single vector
- Vector similarity (distance) $\Rightarrow$ Contextual similarity (distance)

Limitations:

- Unknown words
- More computation necessary

Example Word2Vec Embedding

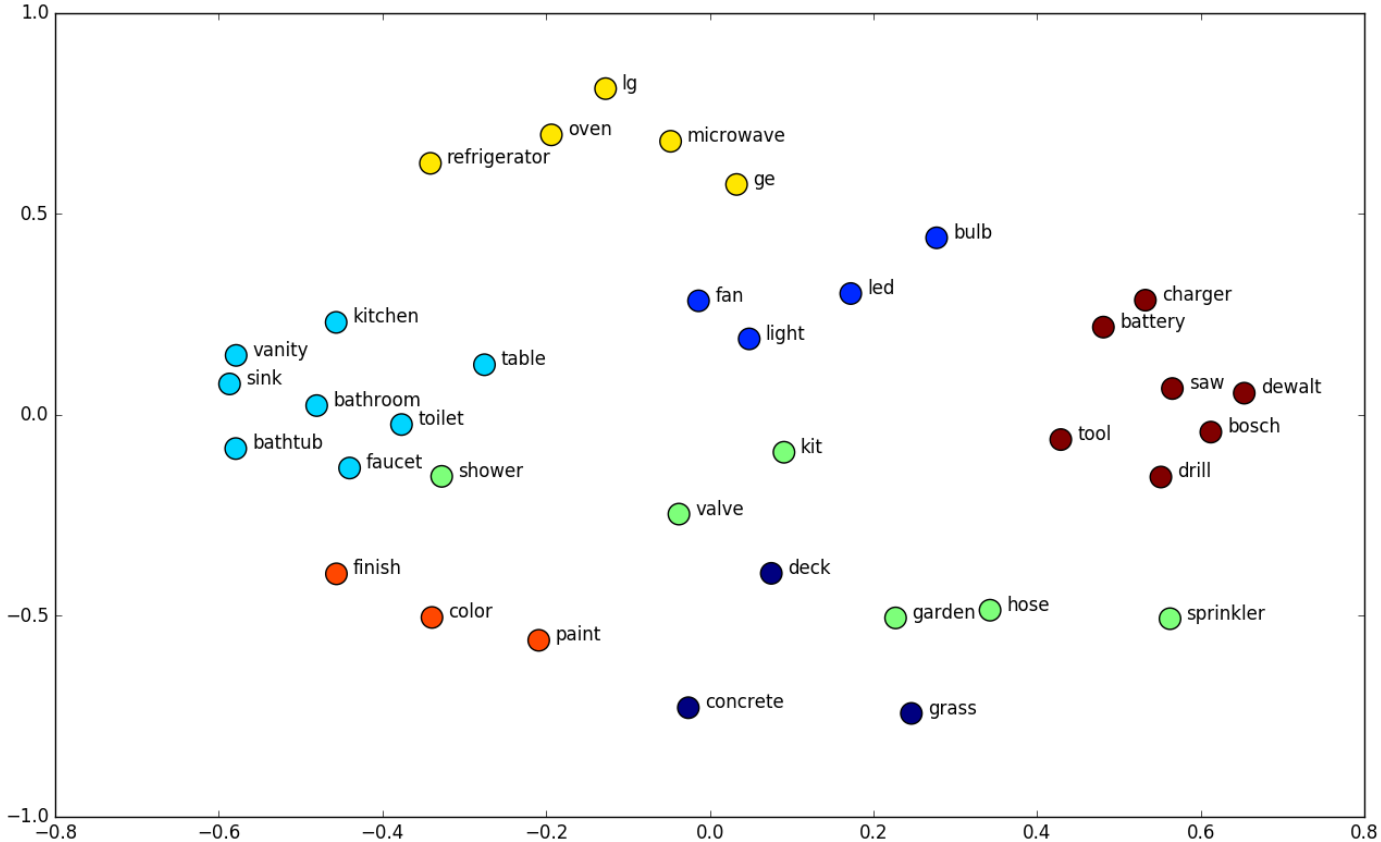

Figure: Clustered Emedding Space from Home Depot Kaggle Competition

Recurrent Neural Networks: Language Modeling to Infinity and Beyond

Recurrent Neural Network

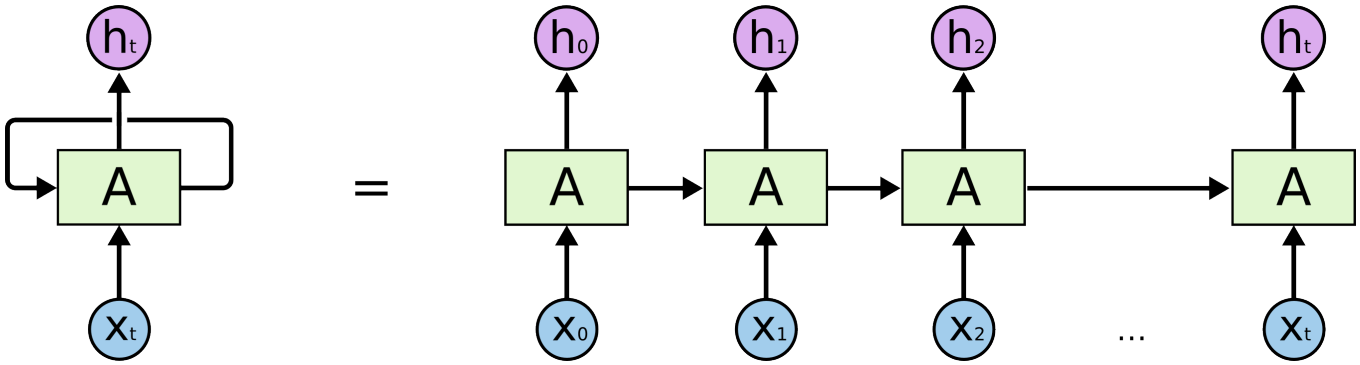

Figure: Simple RNN architecture unrolled over time

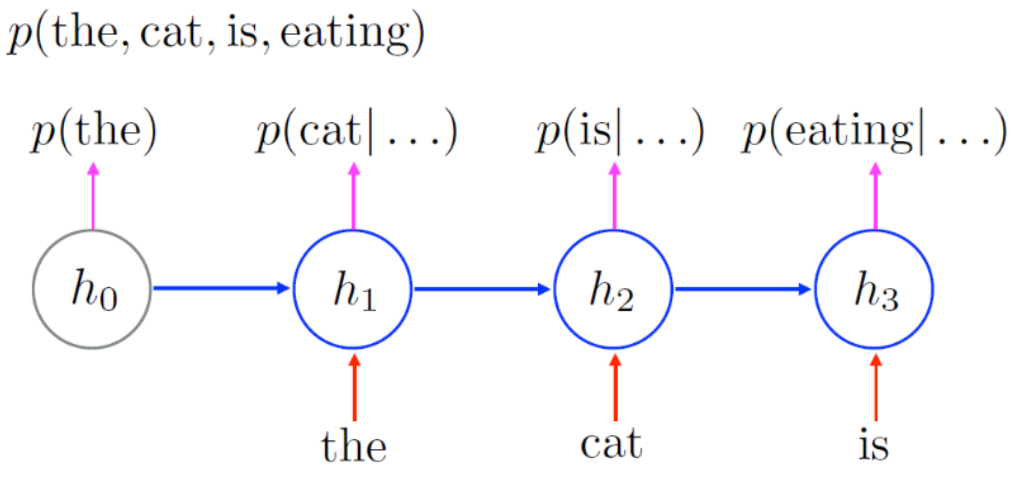

Figure: RNNs *theoretically* can rely on infinite context

RNNs: Expanded Context

As the player shot the ___ball___ *with two* ___minutes___ *to go in the game, the crowd* ___A___ *in amazement.*

A. stares, started, stared (Grammar)

Probabilities (Regularized RNN with Word2Vec):

A. $\frac{p(\text{stared}|\text{shot, the, ball, with, two, minutes, to, go, in, the, game, the, crowd})}{p(\text{stares}|\text{shot, the, ball, with, two, minutes, to, go, in, the, game, the, crowd})} = 1 \Rightarrow$ **???**

What Happened?

"stared" and "stares" are lemmatized to the same root: "stare".

Convolutional Neural Networks: Learning N-Gram Filtersex

1D Convolution with Word2Vec

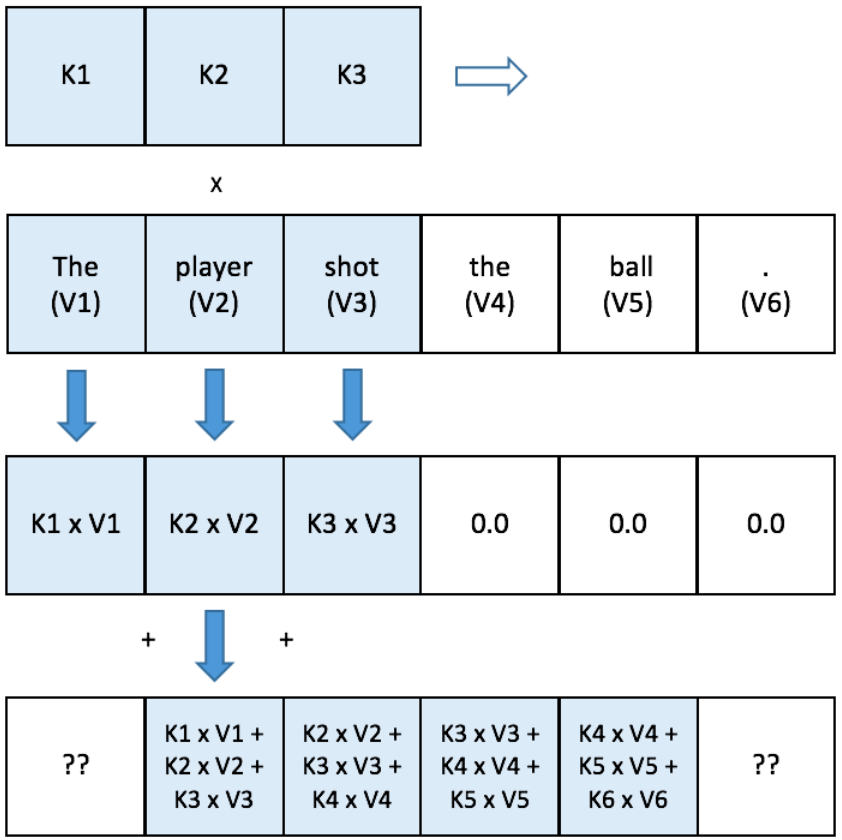

Figure: 1D Convolution with a (l_h=3) filter on a 6-word Sentence.

$$y[n] = x[n] * h[n]$$
$$= \sum_{k=-\infty}^{\infty} x[k] \times h[n-k]$$
$$= \sum_{k=0}^{l_h} x[k] \times h[n-k]$$

- x is the sentence, h is the filter, l_h is the filter length, y is the output.
- h is like a 'phrase stencil' or 'N-gram mask'
- y has larger value when convolved with h's phrase

Convolutional Neural Network (CNN)

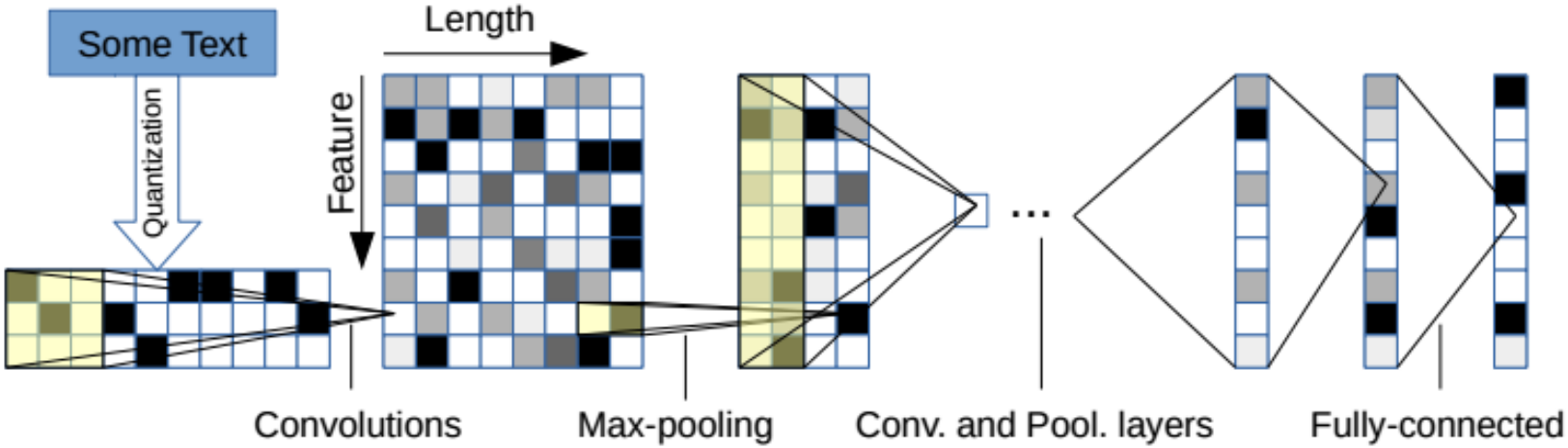

Figure: Training a CNN for Sentiment Analysis (Classification)

Google's State-of-the-Art LM

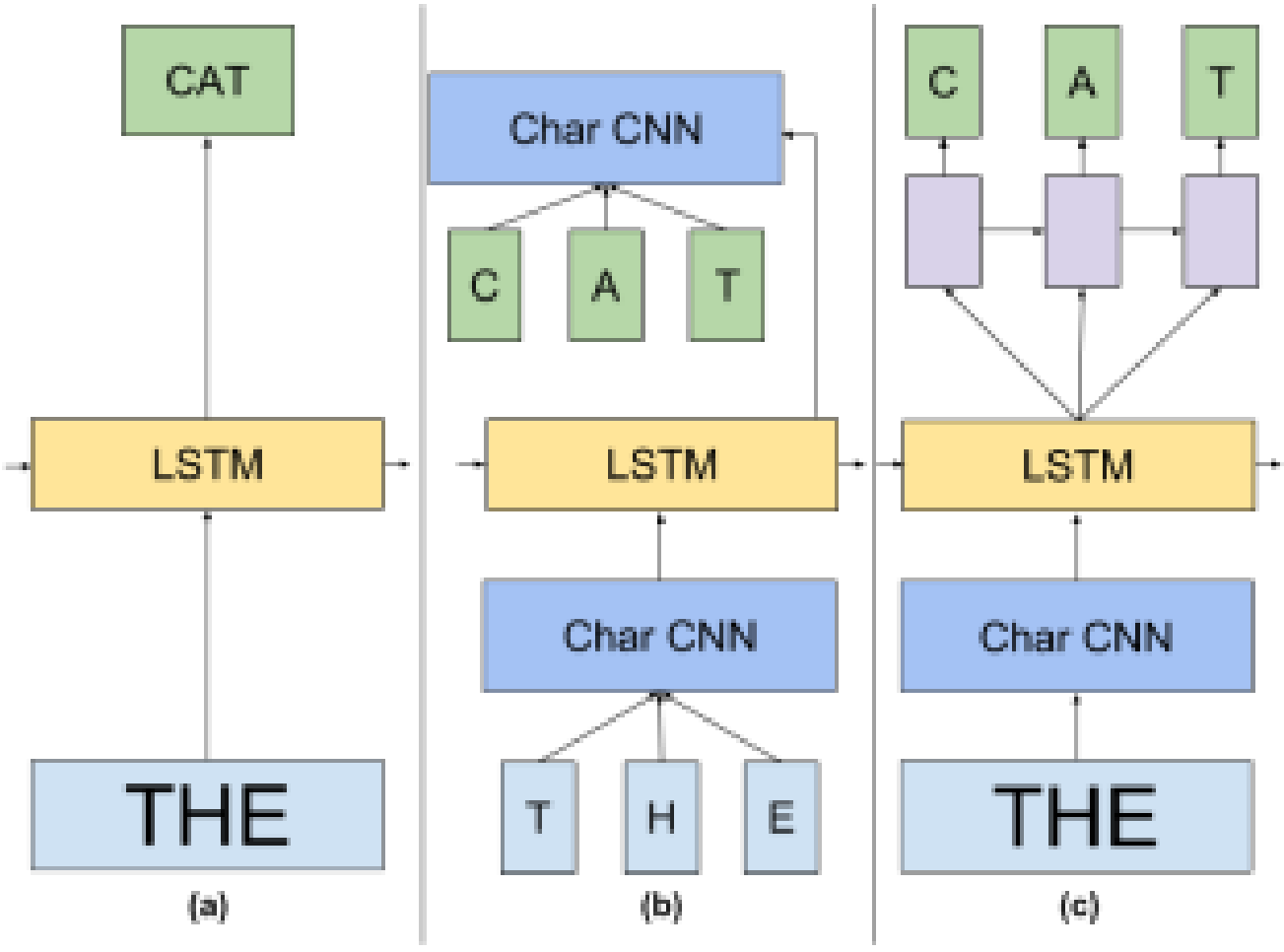

Figure: Char-CNN-LSTM Model (c)

Google's State-of-the-Art LM

Model	Test Perplexity	Number of Params [billions]
Sigmoid-RNN-2048 (Ji et al., 2015a)	68.3	4.1
Interpolated KN 5-gram, 1.1B n-grams (Chelba et al., 2013)	67.6	1.76
Sparse Non-Negative Matrix LM (Shazeer et al., 2015)	52.9	33
RNN-1024 + MaxEnt 9-gram features (Chelba et al., 2013)	51.3	20
LSTM-512-512	54.1	0.82
LSTM-1024-512	48.2	0.82
LSTM-2048-512	43.7	0.83
LSTM-8192-2048 (No Dropout)	37.9	3.3
LSTM-8192-2048 (50% Dropout)	32.2	3.3
2-Layer LSTM-8192-1024 (BIG LSTM)	30.6	1.8
BIG LSTM+CNN Inputs	**30.0**	**1.04**
BIG LSTM+CNN Inputs + CNN Softmax	39.8	**0.29**
BIG LSTM+CNN Inputs + CNN Softmax + 128-dim correction	35.8	**0.39**
BIG LSTM+CNN Inputs + Char LSTM predictions	47.9	**0.23**

Figure: Quanitative Comparisons on 1B Word Benchmark

Takeaways

Main Points:

- Language model outputs must be interpreted as ratios.
- N-Grams are good for quick, localized comparison (synonyms).
- Word2Vec provides a semantic representation of words.
- Neural models are better for text with longer dependencies (sentences, documents).

WS Applications:

- Contextual Synonyms with N-Grams
- Ruled-Based Context Checking with N-Grams
- Sort Sample Sentences with Neural Language Model for User QA

Resources

N-Grams

- http://www.statmt.org/book/slides/07-language-models.pdf
- https://web.stanford.edu/class/cs124/lec/languagemodeling.pdf
- https://research.googleblog.com/2006/08/all-our-n-gram-are-belong-to-you.html

Word2Vec

- http://blog.kaggle.com/2016/05/18/home-depot-product-search-relevance-winners-interview-1st-place-alex-andreas-nurlan/
- https://www.analyticsvidhya.com/blog/2017/05/gpus-necessary-for-deep-learning/

RNN

- http://colah.github.io/posts/2014-07-NLP-RNNs-Representations/
- https://www.linkedin.com/pulse/what-i-learned-from-deep-learning-summer-school-2016-hamid-palangi

CNN

- https://arxiv.org/pdf/1508.06615.pdf
- https://arxiv.org/pdf/1509.01626.pdf